NASA CR-
147824

```
(NASA-CR-147824)  THE RECURSIVE MAXIMUM                N76-27904
LIKELIHOOD PROPORTION ESTIMATOR:  USER'S
GUIDE AND TEST RESULTS (Rice Univ.)  27 p
HC $4.00                             CSCL 09B          Unclas
                                               G3/61   44566
```

ICSA

INSTITUTE FOR COMPUTER SERVICES AND APPLICATIONS

RICE UNIVERSITY

275-025-028

The Recursive Maximum Likelihood Proportion Estimator—User's Guide and Test Results

by

D.L. Van Rooy
Rice University

ABSTRACT:

In this report, we describe our implementation of the recursive maximum likelihood proportion estimator proposed by D. Kazakos in "Recursive Estimation of Prior Probabilities Using the Mixture Approach," (Rice University, ICSA Technical Report #275-025-019). A user's guide to the programs as they currently exist on the IBM 360/67 at LARS, Purdue is included, and test results on LANDSAT data are described. On Hill County data, the algorithm yields results comparable to the standard maximum likelihood proportion estimator.

Institute for Computer Services and Applications
Rice University
Houston, TX 77001
June, 1976

This work was supported under NASA contract NAS 9-12776

I. Introduction:

In this report, we describe our implementation of the recursive maximum likelihood proportion estimator proposed by D. Kazakos in [1]. Numerical results obtained with this algorithm using LANDSAT data are described, and a user's guide for the programs as they currently exist on the IBM 360/67 at LARS (terminal a' ailable at NASA-JSC) is included.

Section II contains a description of the algorithm as implemented. Section III serves as a user's guide to the programs available. In section IV, we describe the numerical results we have obtained with this algorithm. An appendix contains listings of the programs.

II. The Algorithm:

Given a set of n-dimensional measurement vectors $\{x\}$ from M normally distributed multivariate pattern classes H^j, $j = 1, 2, \ldots, M$ the M-1 dimensional recursive maximum likelihood proportion estimate (RMLPE)[1] p^i at the i^{th} data vector is given by

$$p^i = p^{i-1} + \frac{1}{i} L \left[g\, (p^{i-1},\ x_i) \right]^{-1} \left(f_1(x_i) - f_M(x_i)\ ,\ f_2(x_i) - f_M(x_i)\ , \ldots,\ f_{M-1}(x_i) - f_M(x_i) \right) \quad (1)$$

where $f_j(x)$ is the density function for the j^{th} class;

$$f_j(x) = (2\pi)^{-n/2} \; |K_j|^{-\frac{1}{2}} \exp\left[(x-u_j)^T K_j^{-1}(x-u_j)\right] \quad (2)$$

where u_j and K_j are the mean and covariance matrix, respectively, for the j^{th} class; $g(p^{i-1}, x_i)$ is the mixture distribution estimate, i.e.,

$$g(p^{i-1}, x_i) = f_M(x_i) + \sum_{\ell=1}^{m-1} p_\ell^{i-1} \left(f_\ell(x_i) - f_M(x_i)\right) \quad (3)$$

and L is a suitably chosen constant in this approximation. The proportion estimate for the M^{th} class is denoted by p_m^i and given by

$$p_m^i = 1 - \sum_{\ell=1}^{m-1} p_\ell^i \quad (4)$$

In our implementation of this algorithm, we have made several modifications to improve its performance. These include (1) clipping the value of the update (i.e. second) term in eq. (1); (2) renormalizing the p^i at each step so that all $p_j^i \geq 0$ and $\sum_{\ell=1}^{m} p_\ell^i = 1$; and (3) introducing an additional damping term in the update term of eq. (1). The final form of the algorithm is

$$p^i = \mathrm{NORM}\left\{\epsilon,\; p^{i-1} + \frac{1}{i+n_\circ}\, \mathrm{LMT}\left[T, L\, g(p^{i-1}, x_i) \cdot \left(f_1(x_i) - f_M(x_i), f_2(x_i) - f_M(x_i), \ldots, f_{M-1}(x_i) - f_M(x_i)\right)\right]\right\}$$

where $LMT(a, b)$ is the clipping function defined by

$$LMT(a, b) = \tilde{b}$$

with $$\tilde{b}_i = \text{sign}\,(b_i)\ \min\ \left(a,\ |b_i|\right)_j$$

NORM is the renormalizing function defined by

$$NORM(\epsilon,\ y) = \text{the first } M-1 \text{ elements of } \tilde{y}$$

where

$$\tilde{y}_m = 1 - \sum_{i=1}^{m-1} y_i$$

$$\tilde{y}_i \quad y_i$$

If

$$\min\ (\tilde{y}_i) \geq \epsilon > 0 \text{ then finish else}$$

$$\tilde{y}_i \leftarrow \tilde{y}_i - \min_i\ (\tilde{y}_i) + \epsilon \qquad i = 1, 2, \ldots M$$

$$\tilde{y}_i \leftarrow \tilde{y}_i \Big/ \sum_{\ell=1}^{m} \tilde{y}_i \quad ;$$

and $n_\circ$ is a positive constant used to damp out early oscillations of the estimate.

Two other algorithms used in conjunction with this one are (1) an algorithm to calculate an approximation to L and (2) an algorithm to scramble all of the data (the RMLPE uses the stochastic approximation, so the data needs to appear in a random order). The first algorithm calculates the following approximation to L

$$L = \left(u \cdot \min\ |K_j|^{\frac{1}{2}} \right)^{-1}$$

where u is the minimum eigenvalue of H with $H = \left\{ h_{ks} \right\}$ and

$$h_{ks} = (2\pi)^{n/2} \int_{E^n} \left(f_k(x) - f_m(x) \right) \left(f_s(x) - f_m(x) \right) dx$$

$$k, s = 1, 2, \ldots, m$$

The scrambling algorithm employs a procedure described on page 125 of [2].

III. Program Description and Users Guide

Three programs have been written to implement this algorithm: the proportion estimation program, a program to calculate an approximation to L and a program to scramble data prior to estimating proportions. These programs are described below and listings are provided in the appendix.

Proportion Estimation Program:

This program runs on the IBM 360/67 at LARS. Parameters are read from cards describing characteristics of the data and the statistics, the processing to be performed, and the desired outputs. The same data may be processed with several sets of statistics. The data is assumed to be sixteen channel data (from which any subset of channels may be used) residing on file 11 with one logical record per data vector. The data may be labelled or unlabelled. If labelled, the program will calculate the true proportions and print out the means of the estimates along with the associated variances and mean squared error; if unlabelled, the true proportions are read from cards and the same quantities are then computed. Due to the use of the stochastic approximation in this algorithm, the data vectors [illegible] be scrambled before being put on file 11. (Program super[illegible]AM may be used for this purpose.)

The correspondence between the notation used in the preview section and the variables in the program are:

Above	Program
p_i^j	Q(J, *)
$n_\circ$	ISTRT
ϵ	EPS
T	TLM
L	L
G	G
$f_i(x)$	F (a function subprogram)
$\sum_i$	SG
u_i	MU

The programs are set up to handle up to 16 channel data from up to 15 classes with as many as 10 different blocking factors. They can treat an unlimited number of data points. The data enters the program in "lines" which contain ≤ 1500 points.

[N.B. The total number of points need not be an integral multiple of the points per line,

e.g. if there are 5100 total points, we may use NP (the number of points/line = 200 and NL (the number of lines) ≥ 26]

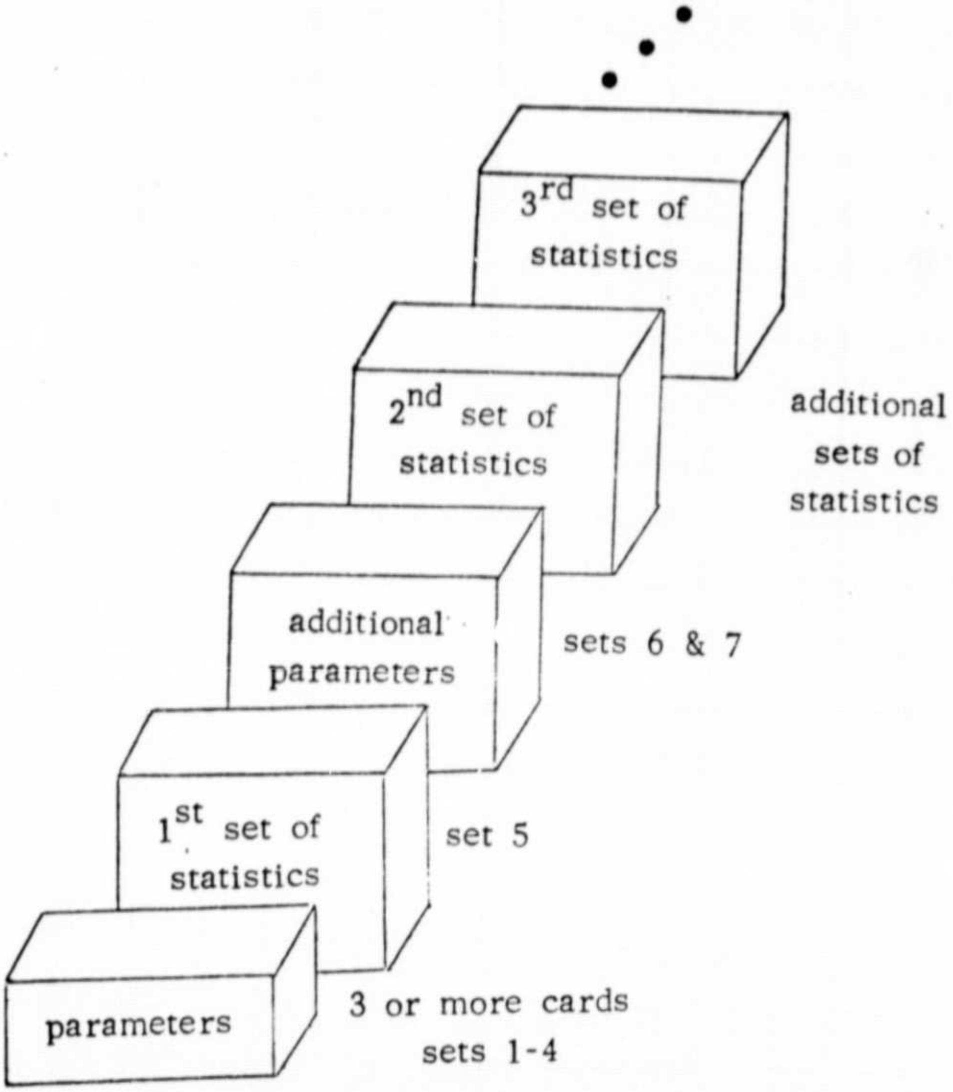

Figure 1

Data Deck Setup for the LANDSAT Version

Figure 1 shows the set-up of the data deck necessary to execute the program. The input parameters and their formats are described below:

1) HEDNG - Title to be printed on the output (20A4)

2) M, MXITER, NK, ISTRT, INQ, ØUTPT, L, TLM, EPS, (K(I), I=1, NK)

(4 I 2, 2 L 1, 3 G 10.8, 10 X, 10 I 3)

M - number of classes used

MXITER - number of sets of statistics to use

NK - number of blocking factors to use (set = 1)

ISTRT - starting value of $n_\circ$ in eq. (1) (default = 99)

INQ - = F if the initial guess for the proportion estimates (Q's) are to be set = $\frac{1}{M}$ (then card set (3) are not used)

= T if the Q's are to be read in (card set (3) is required)

ØUTPT - = T if updated Q's are to be printed after each line of data. Otherwise set = F

L - the L value to be used in eq. (1)

TLM - the maximum permissible absolute value for the update quantity for the Q's (i.e.

$$L \; . \sum_{\ell=1}^{K} \frac{f_j(x_j) - f_m(x_s)}{G(p_{i-1}, x_s)}$$

$$s = K * (i-1) + \ell \,)$$

EPS - minimum allowable value for a Q during the estimation procedure (10^{-2} seems to be a good choice)

K(I), I=1, NK - the blocking factors to be used (set K(1)=1)

3) CSET
(16 L 1)
- $CSET_i = \begin{cases} T & \text{if } i^{th} \text{ channel is to be used} \\ F & \text{otherwise} \end{cases}$

Optional 4) ((Q(I, J), I=1, M-1), J=1, NK)
(16 G 5. 3)

The initial guess for the Q's . Used only if INQ on card 2 is = T

5) CL(I), (MU(J, I), J=1, 16)
(26 X, A 1/(5 X, 5 E 15.8)
(SG (J, I), J=1, 136)
(5 X, 5 E 15.8)

for I=1, 2, . . . M

These cards contain the statistics for the M classes. CL is the class ID, MU, the mean vector, and SG is the covariance matrix stored in symmetric storage mode (i. e. upper triangular part stored by columns). Note that there are 33 cards required for each class. Additional sets of statistics follow card set (7).

6) NP, NL, ØUTPP, ØUTPX, TRUEP
(2 I 5, 3 L 1)

NP - number of points to use per "line" (≤ 1500)

NL - maximum number of "lines" of data

ØUTPP - = T the current true proportions are printed after each line (used only if TRUEP = F)

= F do not print these proportions

ØUTPX - = T print the data vectors

= F do not print the data vectors

TRUEP - = T if the true proportions are to be read in (card set (7) is then required)

= F the class ID is associated with each data vector and the program will calculate the true proportions (card set (7) not used).

Optional 7) (CLS(J), GT(J), J=1, M)

(8(A 2, G 8.6))

CLS(J) - the class ID for the J^{th} class

GT(J) - the true proportions for the J^{th} class

The data vectors should be on file 11 with 1 data vector per logical record in the format

CL, (X(J), J=1, 16)

(8 X, A 1, 6 X, 16 F 4.0, 1 X)

where CL is the class ID (used only if TRUEP on card set 6 = F) and X(J) contains the 16 dimensional data value for a pixel.

The subroutines used in this program are briefly described below:

INSTAT - reads and prints statistics

SUBSET - for LANDSAT data (LD) version, this selects appropriate subsets of the statistics.

F - computes the value of the density function at X

TPØSE - for the pseudo-random (PR) version, transposes the data matrix in situ

GEDATA - obtains or generates a line of data in the required format and order. Also computes the true proportions.

MCHLSK - computes the modified Cholesky decomposition of a covariance matrix stored in symmetric storage mode.

Program to Calculate L:

This program calculates the following approximation to L

$$= \left(h \cdot \min_j \left|\sum_j\right|^{\frac{1}{2}}\right)^{-1}$$

where $h = \min\left(\text{eval}\,(H)\right)$

$$\& \; H_{ks} = (2\pi)^{n/2} \int_{E^n} \left(f_k(x) - f_m(x)\right) \cdot$$

$$\cdot \left(f_s(x) - f_m(x)\right) dx$$

$$k, s, = 1, 2, \ldots, m$$

All notation is as before.

Input parameters to the program are

CSET, M, N

(16L1, 2X, I2, 2X, I2)

[N.B. Our (limited) experience with the proportion estimation algorithm indicates that a value of ~ 3 for L appears optimal despite what this program computes.]

Scrambling Program:

This algorithm scrambles the order of records in a data set and creates a new data set. Two storage arrays are used: one containing the integers 1, 2, . . . , N where N is the total number of records and the other containing space for one data record. A temporary direct access data set, which is the same size as the original data set, is used. The algorithm is described below:

```
1) Set $a_i = i$ for $i = 1, 2, \dots, N$
2) Scramble the elements of the vector $a$.
   (see e.g. ref. [2]).
3) For $i = 1, 2, \dots, N$
     a) Read $i^{th}$ record of original data set
        and store it in vector $d$.
     b) Write $d$ in $a_i^{th}$ record in temporary
        data set.
4) For $i = 1, 2, \dots, N$
     a) Read $i^{th}$ record of temporary data
        set and store it in vector $d$.
     b) Write $d$ on $i^{th}$ record of new data
        set.
5) Finished.
```

Note that step 4 may not be necessary if one can use the data from the temporary direct access data set.

IV. Numerical Results:

A variety of numerical experiments were conducted with this program to determine its characteristic. Both pseudo-random and LANDSAT data were used.

The most significant effect of this algorithm is due to the scrambling (i.e. the order in which the data is input). If the data is not scrambled (i.e. blocks of points from single classes appear to the program) unreliable estimates will be produced. Our experience with LANDSAT data indicates that the entire data set, whose proportions are to be estimated, needs to have the individual pixels scrambled. Various scramblings will produce different estimates with a theoretical variance of L /N where N is the total number of pixels.

Another effect that we noticed was that the variance of the estimate for the M^{th} class was always larger than for other classes. This asymmetry, we feel, is due to the fact that the algorithm estimates proportions for the first M-1 classes, and the estimate for the M^{th} class is then computed as $1 - \sum_{i=1}^{M-1} p^i$. By reordering the classes and then again estimating proportions, it was determined that the variance of the M^{th} class would decrease from ~ 10% to ~ 30%, so the effect may not be too harmful. However, the user should be aware of this and assure that the estimate for the M^{th} class is of the least interest.

Detailed tests of this algorithm were run on some Hill County LANDSAT data in order to compare results with those obtained by Coberly and Odell [3] with five other proportion estimation algorithms. Table 1 shows the results obtained from the recursive maximum likelihood estimator (RMLE) for 2600 pixels of the labelled data as compared to the other five estimators. Note that the RMLE and MLE have almost equal variances and mean squared errors.

Table II shows the results obtained from the RMLE for 8400 pixels of the unlabelled data. Here again the variances and mean squared error are approximately the same as those of the MLE .

V. Conclusions:

Our experience with this algorithm indicates several important factors need be taken account of in using this algorithm: (1) all of the data needs to be scrambled point by point, (2) the class of least importance should be used as the last class, and (3) a value of ~ 3 for the parameter L appears close to optimal.

Our tests indicate that the recursive maximum likelihood estimator(RMLE) produces results of comparable variance and accuracy as the standard maximum likelihood estimator (MLE) of ref. [3]. The amount of computation involved for the RMLE is equivalent to the first iteration of the MLE plus the scrambling of the data. Also, no additional storage is required by this algorithm to store the density functions for each data point.

Further tests of this algorithm with other LANDSAT data will be necessary to determine the effectiveness of this algorithm in the general situation.

Table 1

Summary of Experiment I

(Labeled Data, 2600 Pixels)

		CLASS	ODELL	MLE	RMLE	MIX	MCM	GT
MEAN	WH	.282384	.297041	.300439	.307807	.294353	.274749	.371900
	FA	.265500	.274428	.296764	.277315	.312696	.235257	.286200
	BA	.186089	.174306	.176688	.188004	.183300	.197059	.115400
	GR	.101679	.085066	.085825	.058506	.074200	.075168	.079200
	ST	.164346	.168155	.140178	.168364	.135448	.217765	.147300
VAR	WH	.000038	.000174	.000083	.000094	.000201	.000459	
	FA	.000445	.001385	.000408	.000347	.002216	.003027	
	BA	.000031	.000190	.000084	.000125	.000123	.000288	
	GR	.000080	.000051	.000060	.000050	.000165	.000463	
	ST	.000344	.001829	.000383	.000533	.002376	.004145	
TOTAL VAR		.000937	.003628	.001017	.001149	.005020	.008383	
MSE		.015172	.013322	.010086	.010778	.016572	.032066	

Table 2
Summary of Experiment II
(Total Data Set, 8400 Pixels)

		CLASS	ODELL	MLE	RMLE	MIX	MCM	GT
MEAN	WH	.252733	.262044	.272700	.267325	.226467	.084333	.294000*
	FA	.207300	.183526	.185133	.197484	.353019	.023415	.249000
	BA	.167633	.154997	.151200	.142416	.218460	.325809	.124000
	GR	.186466	.187198	.189166	.158599	.158955	.184282	.138000
	ST	.186066	.212231	.201700	.234173	.043098	.382159	.196000
VAR	WH	.000095	.000263	.000310	.000307	.000232	.000949	
	FA	.000269	.007430	.000472	.000550	.002872	.002114	
	BA	.000063	.000299	.000206	.000220	.000135	.000610	
	GR	.000261	.000363	.000529	.000609	.001042	.000900	
	ST	.000141	.009569	.000832	.000903	.002975	.003402	
TOTAL VAR		.000829	.017924	.002350	.002590	.007257	.007974	
MSE		.008622	.057982	.010273	.008176	.055378	.180347	

REFERENCES

[1] D. Kazakos, "Recursive Estimation of Prior Probabilities Using the Mixture Approach," ICSA Technical Report #275-025-019, Rice University, Houston, Texas, September, 1974.

[2] D. Knuth, Seminumerical Algorithms, the Art of Computer Programming, Vol. 2, Addison-Wesley, Reading, Mass., 1969, p. 125.

[3] W.A. Coberly and P.L. Odell, 'An Empirical Comparison of Five Proportion Estimators," from the annual report of the University of Texas at Dallas for NASA contract NAS 9-13512, January, 1975.

APPENDIX

```
FILE. . .       DVR        FORTRAN  P1

      REAL*4 SG(136,30),MU(16,30),O(30,10),L,X(16,1500),DET(30),
     1 T(30,10)/300*0./
      REAL*4 DUM(2700)
      REAL*4 A(30),G(10)
      REAL*4 OS(30,10),OB(30,10)/300*0./,OV(30,10)/300*0./
      REAL*4 GT(15)
      REAL*4 MSE(30)
      REAL*4 HEDNG (20)
      REAL*8 S,SS
      INTEGER*4 K(10),IP1(10)/10*0/
      INTEGER*4 CHAN(16)
      INTEGER*2 CL(15)
      LOGICAL*1 INO,IENP,IND,OUTPT,END
      LOGICAL*1 CSET(16)
      LOGICAL*1 FRST/.TRUE./
      COMMON /PASS/ SG,MU,M,N
      COMMON /RSET/      NSET,GT
      COMMON /GEPTS/      NL
      COMMON /ORDR/ CL
      NAMELIST /IDAT/ M,  NK,ISTRT,INO,OUTPT,L,TLM,EPS,XMX,MXITER,CSET
      MXCHN=16
      MXPTS=1500
      MXCLS=30
C
C         M - NUMBER OF CLASSES USED (.LE.30)
C         N - NUMBER OF CHANNELS USED (.LE. 16)
C         MXITER - NUMBER OF TIMES TO REDO THE RUN WITH DIFFERENT DATA
C         NK - NUMBER OF K'S TO BE USED
C         ISTRT - INITIAL VALUE OF J IN O(R+1)=O(R)-1/(J+R)*L*F.   (DEF=10
C         INO - LOGICAL VARIABLE INDICATING WHETHER TO READ INITIAL
C                GUESS FOR THE O'S OR NOT
C         OUTPT - LOGICAL VAR =T IF ESTIMATE OF O IS TO BE PRINTED
C                  AFTER EACH LINE OF DATA
C         L - THE L VALUE USED BY THE ALGORITHM
C         TLM - LIMITING VALUE OF L*ABS(FJ-FM)/GI (DEF=INFINITY)
C         EPS - LOWER LIMIT ALLOWED FOR THE O'S
C         XMX - UPPER LIMIT ALLOWED FOR THE O'S
C         CSET - ARRAY INDICATING WHICH OF THE 16 CHANNELS ARE TO BE USED
C         K - THE BLOCKING FACTORS TO BE USED (.LE. 10 OF THEM,EACH LE MP
C         O - THE ESTIMATES OF THE PRIORS
C         SG -  COVARIANCE MATRICES STORED IN SYM STORAGE MODE
C         MU - MEAN VECTORS
C         X - THE DATA VECTORS FOR 1 ''LINE'' OF DATA
C
      PI2=2.*3.14159265
      END=.FALSE.
C
C     READ PARAMETERS
C
      READ (5,1) HEDNG
    1 FORMAT (20A4)
      WRITE (6,4) HEDNG
    4 FORMAT (//1X,20A4,///)
      READ (5,2) M,MXITER,NK,ISTRT,INO,OUTPT,L,TLM,EPS,XMX,(K(I),I=1,NK)
    2 FORMAT (4I2,2L1,4G10.8,10I3)
      READ (5,3) CSET
    3 FORMAT (16L1)
      IF (ISTRT.EQ.0) ISTRT=10
      IF (TLM.LT.1.E-2) TLM=1.E70
      M1=M-1
      ITER=0
      IF (INO) GO TO 5
      GO TO 10
C
C     THEN READ IN INITAAL GUESS FOR PRIORS FOR EACH BLOCKING USED.
C
    5 READ (5,7) ((O(I,J),I=1,M1),J=1,NK)
    7 FORMAT (16G5.3)
      GO TO 20
C
C     ELSE SET INITIAL GUESS FOR PRIORS ALL EQUAL
C
   10 Y=1./M
      DO 15 I=1,M1
      DO 15 J=1,NK
   15 O(I,J)=Y
C
C***********************************************************************
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
C                                                                       DVR00790
   20 CONTINUE                                                          DVR00800
      DO 26 I=1,M)                                                      DVR00810
      DO 26 J=1,NK                                                      DVR00820
   26 OS(I,J)=Q(I,J)                                                    DVR00830
      WRITE (6,IDAT)                                                    DVR00840
      TLM=TLM/L                                                         DVR00850
      WRITE (6,21) (K(I),I=1,NK)                                        DVR00860
   21 FORMAT (' K=',10I5)                                               DVR00870
      DO 23 I=1,NK                                                      DVR00880
   23 WRITE (6,22) (Q(J,I),J=1,M1)                                      DVR00890
   22 FORMAT (' INITIAL Q=',8G16.8)                                     DVR00900
C     GET STATS FOR THE CLASSES                                         DVR00910
C                                                                       DVR00920
   28 CONTINUE                                                          DVR00930
      CALL INSTAT (CSET,  CHAN)                                         DVR00940
      IF (ITER.EQ.0) PI2=PI2**(N/2.)                                    DVR00950
      DO 34 II=1,NK                                                     DVR00960
   34 IP1(II)=0                                                         DVR00970
C                                                                       DVR00980
C     DO CHOLESKY DECOMP OF THE COVARIANCES                             DVR00990
C                                                                       DVR01000
      DO 25 I=1,M                                                       DVR01010
      CALL DCHLSK (SG(1,I),N,X,DET(I))                                  DVR01020
   25 DET(I)=1.D0/(SQRT(DET(I))*PI2)                                    DVR01030
C***********************************************************************DVR01040
C                                                                       DVR01050
C     FETCH ONE LINE OF DATA                                            DVR01060
C                                                                       DVR01070
C***********************************************************************DVR01080
   60 CALL CRDATA(X,NP, CHAN,N,&100)                                    DVR01090
C                                                                       DVR01100
      IF (.NOT.FRST) GO TO 32                                           DVR01110
      FRST=.FALSE.                                                      DVR01120
C                                                                       DVR01130
C     LOOP OVER ALL DATA POINTS TO UPDATE ESTIMATE OF PRIORS            DVR01140
C                                                                       DVR01150
   32 CONTINUE                                                          DVR01160
      IENP=.FALSE.                                                      DVR01170
      DO 30 I=1,NP                                                      DVR01180
      IF (I.EQ.NP) IENP=.TRUE.                                          DVR01190
C                                                                       DVR01200
C     F YIELDS THE VALUE OF THE DENSITY FUNCTION FOR THE CLASS          DVR01210
C                                                                       DVR01220
      FM=F(X(1,I),N,SG(1,M),MU(1,M),DET(M))                             DVR01230
C                                                                       DVR01240
C     THE MIXTURE DISTRIBUTION IS SOORED IN G                           DVR01250
C                                                                       DVR01260
      DO 35 II=1,NK                                                     DVR01270
   35 G(II)=FM                                                          DVR01280
      DO 40 J=1,M1                                                      DVR01290
      FJ=F(X(1,I),N,SG(1,J),MU(1,J),DET(J))                             DVR01300
      A(J)=FJ-FM                                                        DVR01310
      AJ=A(J)                                                           DVR01320
      DO 45 II=1,NK                                                     DVR01330
   45 G(II)=G(II)+Q(J,II)*AJ                                            DVR01340
   40 CONTINUE                                                          DVR01350
C                                                                       DVR01360
C     LOOP TO UPDATE PRIORS FOR EACH BLOCKING FACTOR                    DVR01370
C                                                                       DVR01380
      DO 50 II=1,NK                                                     DVR01390
      IND=.FALSE.                                                       DVR01400
      KI=K(II)                                                          DVR01410
      MIK=MOD(I,KI)                                                     DVR01420
      GI=G(II)                                                          DVR01430
      IF (MIK.EQ.0.OR.IENP) GO TO 52                                    DVR01440
      GO TO 53                                                          DVR01450
C                                                                       DVR01460
C     THEN PREPARE TO UPDATE II-TH PRIORS                               DVR01470
C                                                                       DVR01480
   52 IK=MIN0(MIK,KI)                                                   DVR01490
      IF (IK.EQ.0) IK=KI                                                DVR01500
      IND=.TRUE.                                                        DVR01510
      IP1(II)=IP1(II)+IK                                                DVR01520
   53 CONTINUE                                                          DVR01530
C                                                                       DVR01540
C     COMPUTE UPDATED SUMS                                              DVR01550
C                                                                       DVR01560
```

```
      S=0.
      DO 55 J=1,M1
      XX=A(J)/GI
      T(J,II)=T(J,II)+SIGN(AMIN1(ABS(XX),TLM),XX)
      IF (IND) GO TO 56
      GO TO 55
C
C     UPDATE THE PRIORS AND RESET
C
   56 XX=O(J,II)+L*T(J,II)/(IP1(II)          +ISTRT)
      T(J,II)=0.
      O(J,II)=XX
      S=S+XX
   55 CONTINUE
      IF (.NOT.IND) GO TO 50
C
C     RENORMALIZE THE UPDATED ESTIMATES OF THE PRIORS
C
      O(M,II)=1.DO-S
      S=O(1,II)
      DO 54 J=2,M
      IF (O(J,II).LT.S) S=O(J,II)
   54 CONTINUE
      IF (S.GT.EPS) GO TO 64
      SS=0.
      DO 58 J=1,M
      O(J,II)=O(J,II)-S+EPS
   58 SS=SS+O(J,II)
      DO 62 J=1,M1
   62 O(J,II)=O(J,II)/SS
   64 CONTINUE
   50 CONTINUE
   30 CONTINUE
C***********************************************************************
      IF (.NOT.OUTPT) GO TO 60
C
C     PRINT OUT NEW ESTIMATE OF PRIORS
C
   72 DO 70 II=1,NK
      S=0.
      DO 75 J=1,M1
   75 S=S+O(J,II)
      O(M,II)=1.DO-S
   70 WRITE (6,76) NSET,K(II),(CL(J),O(J,II),J=1,M)
   76 FORMAT (/' UPDATED ESTIMATE OF THE PRIORS FOR LINE ',I5,' WITH BL
     1CKING FACTOR K=',I3/4(' CLASS=',A2,' O=',G15.8,3X))
      IF (.NOT.END) GO TO 60
C
C     UPDATE MEANS AND VARIANCES OF THE ESTIMATES FOR THIS ITERATION.
C
      ITER=ITER+1
      DO 116 II=1,NK
      DO 115 J=1,M
      XX=O(J,II)
      OB(J,II)=OB(J,II)+XX
      OV(J,II)=OV(J,II)+XX*XX
  115 O(J,II)=OS(J,II)
  116 CONTINUE
      NSET=0
      END=.FALSE.
      IF (ITER.LT.MXITER) GO TO 28
C
C     FINISHED WITH ALL DATA, PRINT OUT ESTIMATES & STOP
C
      DO 118 II=1,NK
      S1=0.
      DO 119 J=1,M
      MSE(J)=(OV(J,II)-2.*GT(J)*OB(J,II))/MXITER+GT(J)*GT(J)
      S1=S1+MSE(J)
      XX=OB(J,II)/ITER
      OB(J,II)=XX
      XX=OV(J,II)/ITER-XX*XX
  119 OV(J,II)=SQRT(XX)
      WRITE (6,131) K(II),(CL(J),OB(J,II),OV(J,II),MSE(J),J=1,M)
  131 FORMAT (' K=',I5,' MEANS AND SD''S OF THE ESTIMATES & THE MSE'/
     1 (A3,G16.8,' +-',G16.8,G18.8))
      S1=S1/M
      WRITE (6,121) S1
  118 CONTINUE
  121 FORMAT (' ***  MEAN MSE',G16.8)
      STOP
C
C     FINISHED WITH ALL DATA FOR THIS ITERATION
C
  100 END=.TRUE.
      GO TO 72
      END
```

```
      FUNCTION F(X,N,L,MU,DET)
C
C     COMPUTE THE VALUE OF THE DENSITY FUNCTION AT X
C
      REAL*4 X(1),L(1),MU(1),Y(16)
      REAL*8 TF,S
C
C     SOLVE L Y=X-MU WHERE L IS THE CHOLESKY DECOMP OF COVAR MATRIX.
C     DIAG ELEMENTS OF L ARE STORED AS RECIPROCALS.
C
      S=X(1)-MU(1)
      Y(1)=S
      TF=S*S*L(1)
      IF (N.EQ.1) GO TO 15
      K=1
C
C     LOOP TO COMPUTE Y(I'S)
C
      DO 10 I=2,N
      S=X(I)-MU(I)
      JJ=I-1
      DO 20 J=1,JJ
      K=K+1
   20 S=S-L(K)*Y(J)
      K=K+1
      Y(I)=S
      TF=TF+S*S*L(K)
   10 CONTINUE
   15 CONTINUE
      IF (TF.LT.325.) GO TO 17
      F=0.
      RETURN
   17 F=EXP(SNGL(-TF/2.))*DET
      RETURN
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      SUBROUTINE INSTAT (CSET, CHAN)                                    DVR02790
      REAL*4 SG(136,30),MU(16,30)                                       DVR02800
      INTEGER*4 CHAN(1)                                                 DVR02810
      INTEGER*2 CL(15),NC(15),NP(15)                                    DVR02820
      LOGICAL*1 CSET(16)                                                DVR02830
      COMMON /PASS/ SG,MU,M,N                                           DVR02840
      COMMON /ORDR/ CL                                                  DVR02850
      DO 5 I=1,M                                                        DVR02860
      READ (5,1) CL(I),(MU(J,I),J=1,16)                                 DVR02870
    1 FORMAT (26X,A1/(5X,5F15.8))                                       DVR02880
    5 READ (5,4) (SG(J,I),J=1,136)                                      DVR02890
    4 FORMAT (5X,5F15.8)                                                DVR02900
      CALL SUBSET(CSET,CHAN)                                            DVR02910
      DO 10 I=1,M                                                       DVR02920
      WRITE (6,2) CL(I),NC(I),NP(I),(MU(J,I),J=1,N)                     DVR02930
    2 FORMAT (//' CLASS',A2,I5,' NO. OF PTS=',I5/' MEAN',10F11.4/       DVR02940
     1 1X,6F11.4)                                                       DVR02950
      WRITE (6,3)                                                       DVR02960
    3 FORMAT (' COVARIANCE')                                            DVR02970
      J1=1                                                              DVR02980
      J2=0                                                              DVR02990
      DO 20 J=1,N                                                       DVR03000
      J2=J2+J                                                           DVR03010
      J1=J1+J-1                                                         DVR03020
   20 WRITE (6,21) (SG(L,I),L=J1,J2)                                    DVR03030
   21 FORMAT (/(1X,13F10.4))                                            DVR03040
   10 CONTINUE                                                          DVR03050
      RETURN                                                            DVR03060
      END                                                               DVR03070
```

```
      SUBROUTINE SUBSET (CSET,  CHAN)                                   DVR03080
      LOGICAL*1 CSET(1)                                                 DVR03090
      REAL*4 SG(136,30),MU(16,30)                                       DVR03100
      INTEGER*4 CHAN(1)                                                 DVR03110
      COMMON /PASS/ SG,MU,M,K                                           DVR03120
      ISB(I,J)=(I*(I-1))/2+J                                            DVR03130
      IDAG(I)=(I*(I+1))/2                                               DVR03140
C                                                                       DVR03150
C     FIND CHANNELS DESIRED                                             DVR03160
C                                                                       DVR03170
      K=0                                                               DVR03180
      DO 10 I=1,16                                                      DVR03190
      IF (.NOT.CSET(I)) GO TO 10                                        DVR03200
      K=K+1                                                             DVR03210
      CHAN(K)=I                                                         DVR03220
   10 CONTINUE                                                          DVR03230
C                                                                       DVR03240
C     SELECT APPROPRIATE SUBSETS OF SG & MU                             DVR03250
C                                                                       DVR03260
      JJ=0                                                              DVR03270
      DO 20 I=1,K                                                       DVR03280
C                                                                       DVR03290
C     STORE DIAGONAL ELEMENTS                                           DVR03300
C                                                                       DVR03310
      JJ=JJ+1                                                           DVR03320
      DO 25 L=1,M                                                       DVR03330
      SG(JJ,L)=SG(IDAG(CHAN(I)) ,L)                                     DVR03340
   25 MU(I,L)=MU(CHAN(I),L)                                             DVR03350
      IF (I.EQ.K) RETURN                                                DVR03360
C                                                                       DVR03370
C     MOVE ALL ELEMENTS OF NEXT ROW EXCDPT THE DIAGONAL ONE             DVR03380
C                                                                       DVR03390
      L1=CHAN(I+1)                                                      DVR03400
      DO 30 J=1,I                                                       DVR03410
      L2=CHAN(J)                                                        DVR03420
      JJ=JJ+1                                                           DVR03430
      DO 30 L=1,M                                                       DVR03440
   30 SG(JJ,L)=SG(ISB(L1,L2),L)                                         DVR03450
   20 CONTINUE                                                          DVR03460
      STOP                                                              DVR03470
      END                                                               DVR03480
      SUBROUTINE GEDATA (X,NP.CHAN,KK,*)
      REAL*4 SG(136,30),MU(16,30)
      REAL*4 X(16,1500),GT(15)
      INTEGER*4 CHAN(1)
      INTEGER*2 CL,CLS(15),PTS(15)/15*0/,ITER/0/,SCLS(15)
      INTEGER*2 LPTS(15)/15*0/,IPTS(15)/15*0/
      LOGICAL*1 FRST/.TRUE./
      LOGICAL*1 OUTPP,OUTPX
      LOGICAL*1 TRUEP
C
C     FOR CUBERLY'S DATA
C
      COMMON /PASS/ SG,MU,M
      COMMON /RSET/ LINE,GT
      COMMON /ORDR/ SCLS
      COMMON /GEPTS/       NL
      IF (.NOT.FRST) GO TO 10
      FRST=.FALSE.
      LINE=0
      K=0
C
C       NP - NUMBER OF POINTS PER ''LINE'' (.LE.1500)
C       NL - NUMBER OF LINES
C       OUTPP=T - PRINT RUNNING TRUE PROPORTIONS
C       OUTPX=T - PRINT DATA VECTORS
C     TRUEP=T - READ IN TRUE PROPORTIONS
C
      READ (5,1) NP,NL,OUTPP,OUTPX,TRUEP
    1 FORMAT (2I5,3L1)
      NPS=NP
      WRITE (6,2) NP,NL,OUTPP,OUTPX,TRUEP
    2 FORMAT (' NP=',I5,' NL=',I5,' OUTPP=',L8,' OUTPX=',L8,' TRUEP=',
     1 L8)
      IF (TRUEP) READ (5,3) (CLS(J),GT(J),J=1,M)
    3 FORMAT (8(A2,G8.6))
   10 LINE=LINE+1
      NP=NPS
      IF (LINE.LE.NL) GO TO 20
C
C     FINISHED WITH THIS PASS OF THE DATA
C
      REWIND 11

      ITER=ITER+1
      IF (ITER.GT.1) RETURN 1
C
C     COMPUTE TRUE PROPORTIONS & REARRANGE CLASSES TO HHOSE IN STATS
C
      IF (TRUEP) K=M
      JJ=0
      DO 50 I=1,K
      IF (TRUEP) PTS(I)=0
      DO 55 J=1,K
      IF (CLS(J).EO.SCLS(I)) GO TO 52
   55 CONTINUE
      WRITE (6,53) SCLS(I)
   53 FORMAT (' CLASS NOT FOUND',A3)
      GO TO 50
   52 IF (I.EO.J) GO TO 50
      L=CLS(I)
      CLS(I)=CLS(J)
      CLS(J)=L
      IF (TRUEP) GO TO 50
      L=PTS(I)
      PTS(I)=PTS(J)
      PTS(J)=L
   50 CONTINUE
      IF (TRUEP) GO TO 57
      DO 54 I=1,K
      JJ=JJ+PTS(I)
   54 CONTINUE
C
C     PRINT OUT PROPORTIONS
C
   57 CONTINUE
      XJ=JJ
      WRITE (6.51)
   51 FORMAT ('1 DATA OBSERVED'/' CLASS',T10,'POINTS',T20,'PROPORTIONS')
      DO 60 J=1,K
      IF (TRUEP) GO TO 60
      XC=PTS(J)/XJ
      GT(J)=XC
   60 WRITE (6,61) CLS(J),PTS(J),GT(J)
   61 FORMAT (A3,T10,I6,T20,G16.8)
      RETURN 1
C ****************************************************************
   20 CONTINUE
      DO 30 I=1,NP
C
C     READ OBSERVATION VECTOR
C
      READ (11,31,END=35,ERR=35) CL, (X(J,I),J=1,16)
C
   31 FORMAT (8X,A1,6X,16F4.0)
C     SELECT SUBSET OF CHANNELS DESIRED
C
      DO 37 L=1,KK
   37 Y(L,I)=X(CHAN(L),I)
      IF (TRUEP) GO TO 30
      IF (ITER.GE.1) GO TO 30
      IF (K.EQ.0) GO TO 42
C
C     TALLY FOR COMPUTING TRUE PROPORTIONS
      DO 40 J=1,K
      IF (CL.EQ.CLS(J)) GO TO 45
   40 CONTINUE
   42 K=K+1
      J=K
      CLS(K)=CL
   45 PTS(J)=PTS(J)+1
   30 CONTINUE
      IF (ITER.GE.1) RETURN
   64 CONTINUE
      IF (.NOT.OUTPP.OR.TRUEP) GO TO 87
C
C     COMPUTE PROPORTIONS OF DATA FOR THIS LINE AND TO DATE
C
      L=0
      LT=0
      DO 80 J=1,K
      LT=LT+PTS(J)
      LPTS(J)=PTS(J)-IPTS(J)
      IPTS(J)=PTS(J)
   80 L=L+LPTS(J)
      XX=L
      DO 85 J=1,K
      PROP=LPTS(J)/XX
      TPRO=PTS(J)/FLOAT(LT)
   85 WRITE (6,86) LINE,CLS(J),PROP,TPRO
   86 FORMAT (' LINE',I5,' CLASS ',A1,' TRUE PROPORTIONS FOR THIS LINE =
     1',G16.8,' TOTAL TRUE PROPOR. TO DATE =',G16.8)
C
C     WRITE OUT OBSERVATION VECTORS
C
   87 CONTINUE
      IF (.NOT.OUTPX) RETURN
      WRITE (6,66)
   66 FORMAT ('1  X=')
      DO 70 I=1,NP
      L=I+(LINE-1)*NPS
   70 WRITE (6,62) L,(X(J,I),J=1,KK)
   62 FORMAT (I5,16F5.0)
      RETURN
   35 WRITE (6,36) LINE,I
   36 FORMAT (' END OF DATA ON LINE',I5,' AND PIXEL',I5)
      REWIND 11
      NP=I-1
      LINE=NL
      IF (NP.LE.0) GO TO 10
      IF (ITER.LT.1) GO TO 64
      RETURN
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      SUBROUTINE MCHLSK(KK,NV,DUM,DET)
C
C*************************************************************************
C         THIS ROUTINE COMPUTES THE MODIFIED CHOLESKY DECOMPOSITION OF
C         THE COVARIANCE MATRIX.  THE DECOMPOSITIONS OVERLAY THE ELEMENTS
C         OF THE COVARIANCE MATRIX.
C         KK=L D L*
C*************************************************************************
C
C         KK - THE COVARIANCE MATRIX STORED IN SYMMETRIC STORAGE MODE.
C         NV - THE NUMBER OF CHANNELS USED
C         DUM - A WORK AREA OF SIZE NV-1
C         DET - THE DETERMINANT OF THE COVARIANCE MATRIX.
C
      REAL KK(1),DUM(1)
      LOGICAL*1 JE1
      JE1=.TRUE.
      J1=0
      JD=0
      DET=1.
C
C         LOOP OVER ALL CHANNELS
C
      DO 10 J=1,NV
      KL=J-1
      L=J+1
      JD=J1
      J1=J1+J
      TF=KK(J1)
      IF (JE1) GO TO 12
      K1=0
C
C         COMPUTE THE DIAGONAL ELEMENTS OF D AND STORE IN KK
C         TEMPORARILY STORE THE PRODUCT KK(I,I)*KK(J,I) IN DUM(I)
C
      DO 15 I=1,KL
      R=KK(JD+I)
      K1=K1+I
      R1=KK(K1)*R
      TF=TF-R1*R
      DUM(I)=R1
   15 CONTINUE
      KK(J1      )=TF
   12 CONTINUE
      DET=DET*TF
      IF (L.GT.NV) GO TO 10
      IRD=J1-L+1
C
C         COMPUTE THE R,J-TH ELEMENT OF L USING T1
C
      DO 20 IR=L,NV
      IRD=IRD+IR-1
      T1=KK(IRD+J)
      IF (JE1) GO TO 16
      DO 25 I=1,KL
      T1=T1-DUM(I)*KK(IRD+I)
   25 CONTINUE
   16 KK(IRD+J)=T1/TF
   20 CONTINUE
      JE1=.FALSE.
   10 CONTINUE
      J1=0
C
C         STORE THE ELEMENTS OF D IN THIS FORM FOR USE IN SUBROUTINE
C           CLASS
C
      DO 30 J=1,NV
      J1=J1+J
   30 KK(J1)= 1./KK(J1)
      RETURN
      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
LEVEL 21.8 ( JUN 74 )                               OS/360  FORTRAN H                                                DATE  76.

           COMPILER OPTIONS - NAME=  MAIN,OPT=00,LINECNT=60,SIZE=0000K,
                              SOURCE,EBCDIC,NOLIST,NODECK,LOAD,MAP,NOEDIT,NOID,XREF
                 C
                 C      THIS PROGRAM SCRAMBLES NP RECORDS (.LE.10000) EACH OF LENGTH 19
                 C      WORDS CONTAINED ON FILE 11 AND PUTS THE RESULTS ON FILE 12
                 C
  ISN 0002              INTEGER*2 INT(10000),DAT(19)
  ISN 0003              DEFINE FILE 8(10000,10,U,IJ)
  ISN 0004              ISEED=1314159793
  ISN 0005              READ (5,11) NP
  ISN 0006           11 FORMAT (I5)
                 C
                 C      GENERATE THE INTEGERS 1,2,....,NP AND STORE IN ARRAY INT
                 C
  ISN 0007              DO 20 I=1,NP
  ISN 0008           20 INT(I)=I
                 C
                 C      SCRAMBLE ARRAY INT
                 C
  ISN 0009              J=NP
                 C
                 C         GGUBF GENERATES A RANDOM NUMBER FROM U(0,1)
                 C
  ISN 0010           25 R=GGUBF(ISEED)
  ISN 0011              IX=J*R+1
  ISN 0012              IT=INT(J)
  ISN 0013              INT(J)=INT(IX)
  ISN 0014              INT(IX)=IT
  ISN 0015              J=J-1
  ISN 0016              IF (J.GT.1) GO TO 25
  ISN 0018              WRITE (6,7)
  ISN 0019            7 FORMAT (' SHFLD')
                 C
                 C      LOOP TO PUT I-TH RECORD IN INT(I)-TH POSITION ON FILE 8 (A TEMP.
                 C      DIRECT ACCESS FILE)
                 C
  ISN 0020              DO 30 I=1,NP
  ISN 0021              L=INT(I)
  ISN 0022              FIND(8'L)
  ISN 0023              READ (11,1) DAT
  ISN 0024            1 FORMAT (2I3,2A1,16I3)
  ISN 0025           30 WRITE (8'L) DAT
  ISN 0026              WRITE (6,8)
  ISN 0027            8 FORMAT (' ON 8')
                 C
                 C      COPY FILE 8 TO FILE 12
                 C
  ISN 0028              DO 40 I=1,NP
  ISN 0029              I1=I+1
  ISN 0030              READ (8'I) DAT
  ISN 0031              FIND(8'I1)
  ISN 0032              IF (I/50*50.EQ.I) WRITE (6,3) I
  ISN 0034            3 FORMAT (I8)
  ISN 0035           40 WRITE (12,1) DAT
  ISN 0036              STOP
  ISN 0037              END
```